

COMPTE RENDU

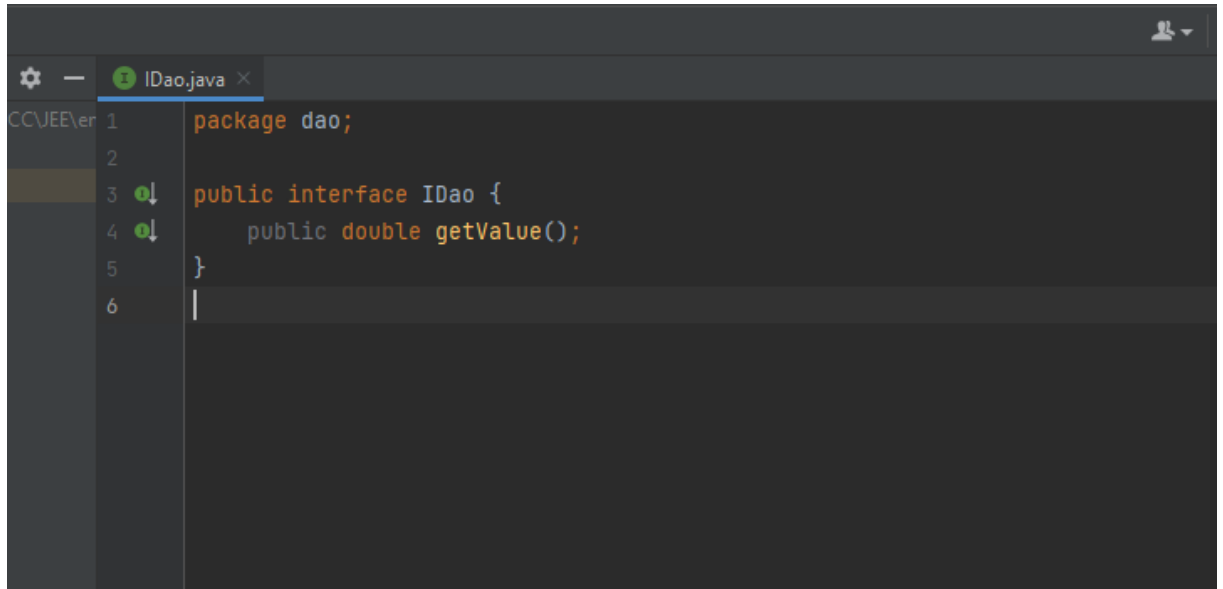
TP1 JEE : Inversion de contrôle et Injection des dépendances

Par : **Assimi DIALLO**

Encadrant : **M. YOUSSEFI**

Par instanciation statique et dynamique

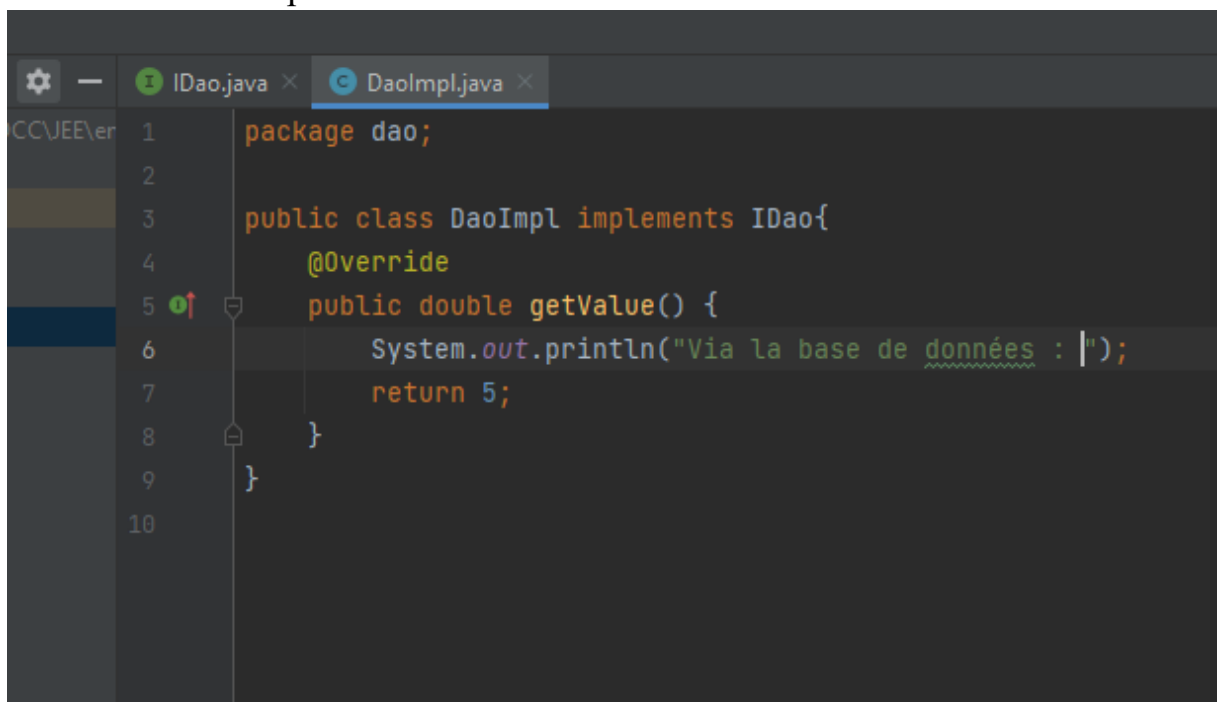
1- Création de l'interface IDao



The screenshot shows an IDE window with a tab for IDao.java. The code defines a package 'dao' and a public interface 'IDao' with a single method 'public double getValue()'. The line numbers 1 through 6 are visible on the left margin.

```
1 package dao;
2
3 public interface IDao {
4     public double getValue();
5 }
6
```

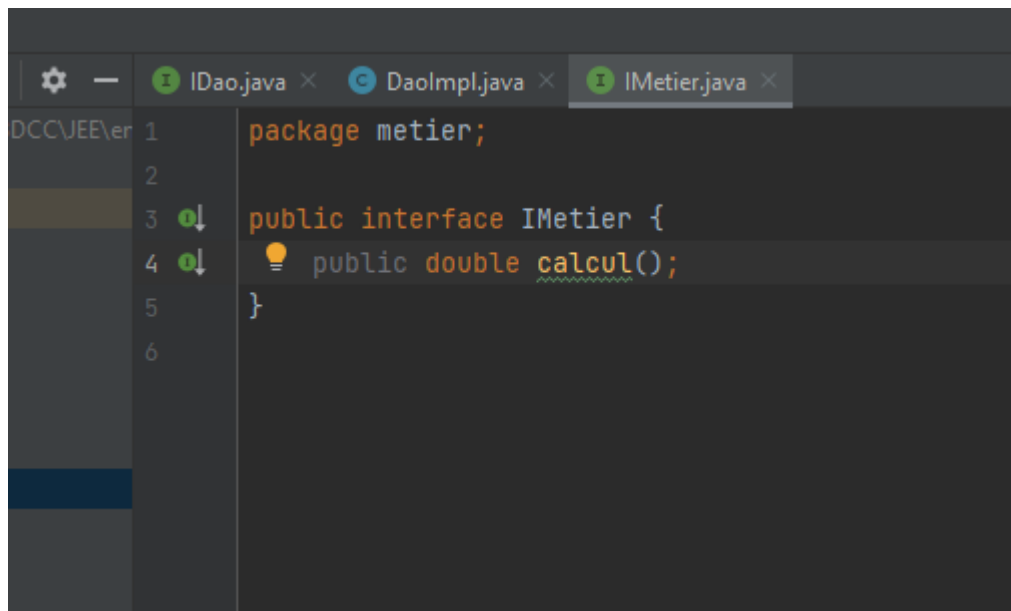
2- Création d'une implémentation de l'interface IDao



The screenshot shows an IDE window with two tabs: IDao.java and DaoImpl.java. The DaoImpl.java tab is active, showing a package 'dao' and a public class 'DaoImpl' that implements 'IDao'. The class contains an '@Override' annotation and a 'public double getValue()' method that prints 'Via la base de données : |' and returns 5. The line numbers 1 through 10 are visible on the left margin.

```
1 package dao;
2
3 public class DaoImpl implements IDao{
4     @Override
5     public double getValue() {
6         System.out.println("Via la base de données : |");
7         return 5;
8     }
9 }
10
```


3- Création de l'interface IMetier



The screenshot shows an IDE with three tabs: IDao.java, DaoImpl.java, and IMetier.java. The IMetier.java tab is active, displaying the following code:

```
1 package metier;
2
3 public interface IMetier {
4     public double calcul();
5 }
6
```

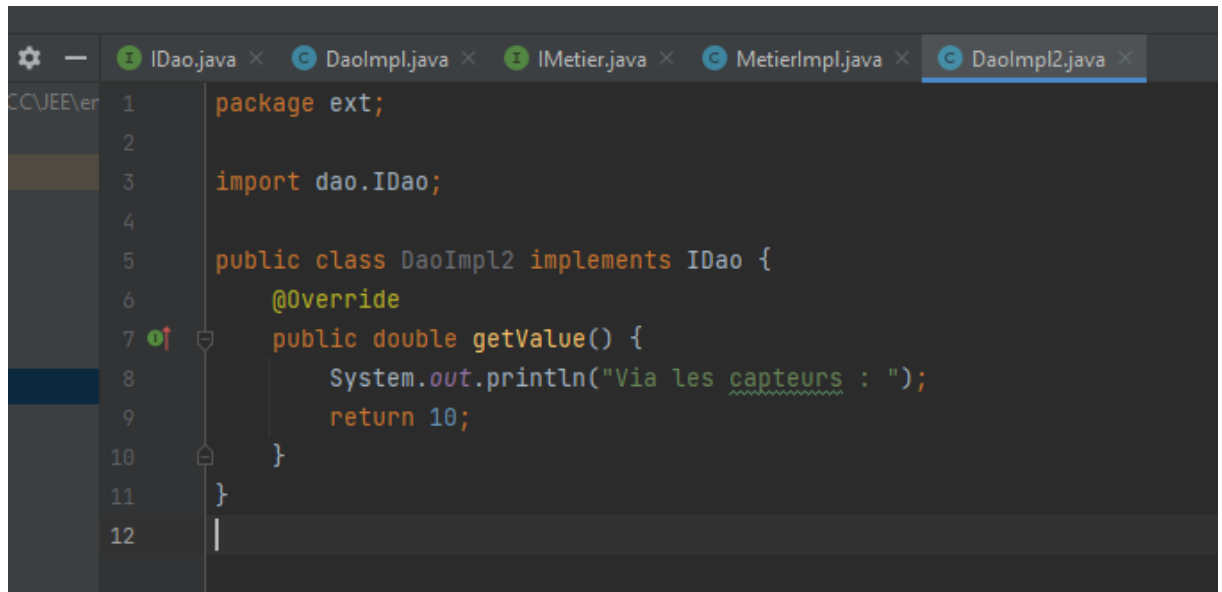
4- Création d'une implémentation de cette interface en utilisant le couplage faible



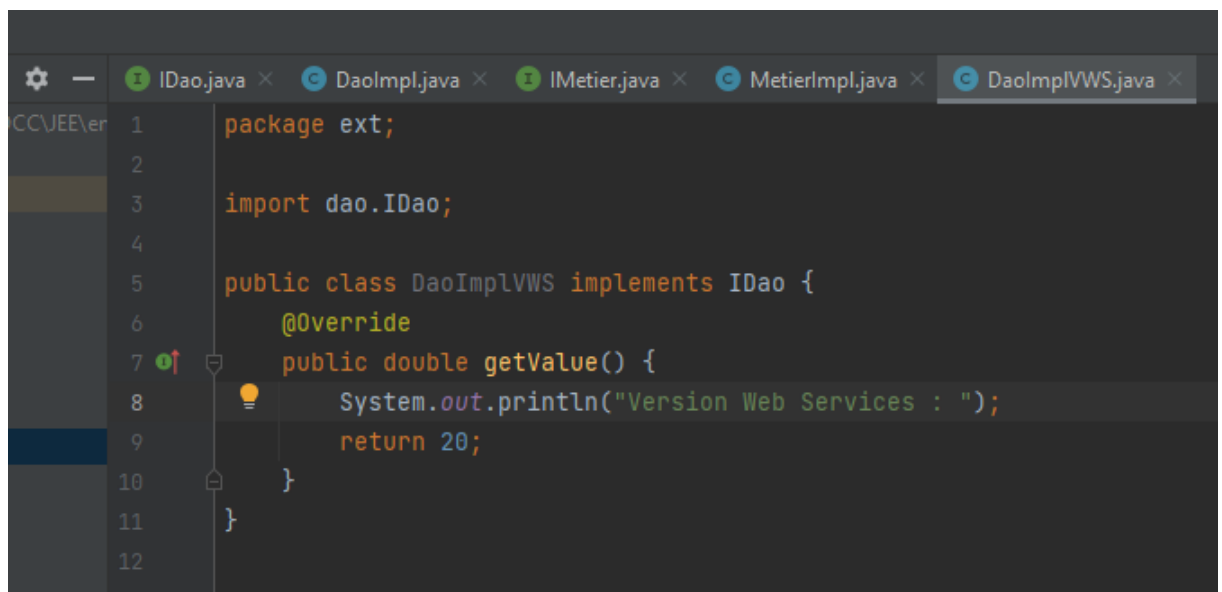
The screenshot shows an IDE with four tabs: IDao.java, DaoImpl.java, IMetier.java, and MetierImpl.java. The MetierImpl.java tab is active, displaying the following code:

```
1 package metier;
2
3 import dao.IDao;
4
5 public class MetierImpl implements IMetier{
6     private IDao dao;
7
8     public void setDao(IDao dao) { this.dao = dao; }
9
10
11
12     @Override
13     public double calcul() {
14         double nb = dao.getValue();
15         return 2*nb;
16     }
17 }
```

Nous allons créer 2 implémentations autres implémentations de l'interface IDao. Nous les mettrons dans un autre package que nous allons nommer **ext**.



```
1 package ext;
2
3 import dao.IDao;
4
5 public class DaoImpl2 implements IDao {
6     @Override
7     public double getValue() {
8         System.out.println("Via les capteurs : ");
9         return 10;
10    }
11 }
12
```



```
1 package ext;
2
3 import dao.IDao;
4
5 public class DaoImplVWS implements IDao {
6     @Override
7     public double getValue() {
8         System.out.println("Version Web Services : ");
9         return 20;
10    }
11 }
12
```

5- Faisons l'injection des dépendances :

a- Par instanciation statique

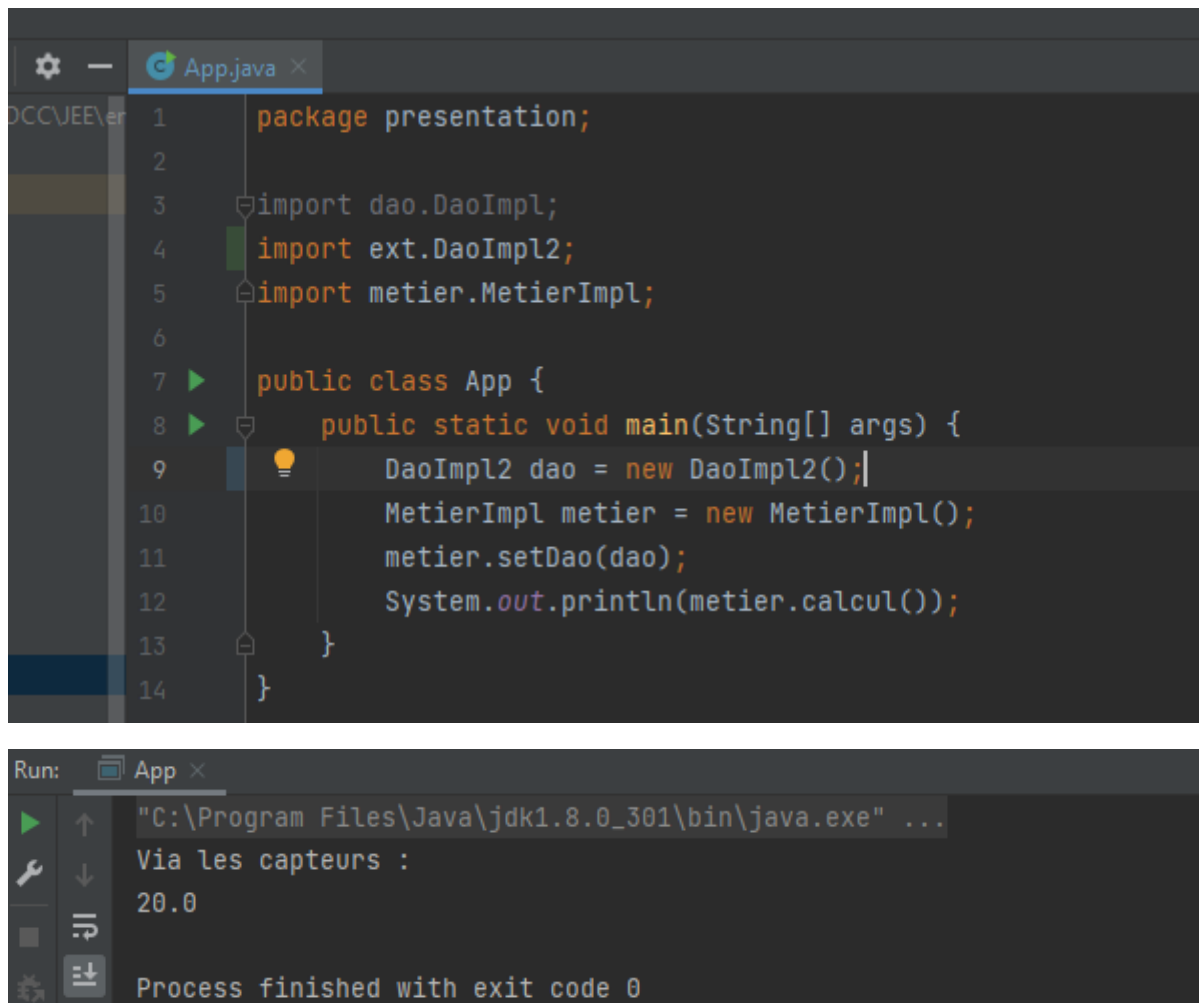
Pour cela nous allons créer une classe **App** dans un package **Présentation**, classe dans laquelle nous allons utiliser la méthode **main**.

```
1 package presentation;
2
3 import dao.DaoImpl;
4 import metier.MetierImpl;
5
6 public class App {
7     public static void main(String[] args) {
8         DaoImpl dao = new DaoImpl();
9         MetierImpl metier = new MetierImpl();
10        metier.setDao(dao);
11        System.out.println(metier.calcul());
12    }
13 }
```

```
Run: App x
"C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ...
Via la base de données :
10.0
Process finished with exit code 0
```

Ici si nous choisissons d'utiliser une des versions, des implémentations de l'interface IDao, nous devons modifier la ligne 8. Le code n'est donc pas fermé à la modification bien qu'ouvert à l'extension.

Comme on peut le voir ci-dessous :



The image shows a screenshot of an IDE with two panels. The top panel displays the source code of a Java file named `App.java`. The code is as follows:

```
1 package presentation;
2
3 import dao.DaoImpl;
4 import ext.DaoImpl2;
5 import metier.MetierImpl;
6
7 public class App {
8     public static void main(String[] args) {
9         DaoImpl2 dao = new DaoImpl2();
10        MetierImpl metier = new MetierImpl();
11        metier.setDao(dao);
12        System.out.println(metier.calcul());
13    }
14 }
```

The bottom panel shows the output of running the application. The command executed is `"C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ...`. The output is:

```
Via les capteurs :
20.0
Process finished with exit code 0
```

b- Par instanciation dynamique

Nous allons donc passer par une instanciation dynamique pour rendre le code fermé à la modification. Pour cela nous allons créer un fichier de configuration dans lequel nous pourrions préciser l'implémentation IDao que nous souhaitons utiliser. Nous allons définir une classe **App2** dans laquelle nous allons utiliser la méthode **main**.

```
App2.java
4      import metier.IMetier;
5
6      import java.io.File;
7      import java.lang.reflect.Method;
8      import java.util.Scanner;
9
10     public class App2 {
11     public static void main(String[] args) {
12     try {
13         Scanner scanner= new Scanner( new File( pathname: "config.txt"));
14
15         String daoClassname = scanner.nextLine();
16         Class cdao = Class.forName(daoClassname);
17         IDao dao = (IDao) cdao.newInstance();
18
19         String metierClassname = scanner.nextLine();
20         Class cmetier = Class.forName(metierClassname);
21         IMetier metier = (IMetier) cmetier.newInstance();
22
23         Method meth = cmetier.getMethod( name: "setDao", IDao.class);
24         meth.invoke(metier, dao);
25         System.out.println(metier.calcul());
26     } catch (Exception e){
27         e.printStackTrace();
28     }
29     }
30 }
```

```
config.txt
1      dao.DaoImpl
2      metier.MetierImpl
```

Il s'affichera :

```
Run: App2
"C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ...
Via la base de données :
10.0
Process finished with exit code 0
```

Modifier le fichier de configuration **config.txt** pour utiliser la version Services Web :

The image shows two screenshots from an IDE. The top screenshot displays the 'config.txt' file with the following content:

```
1 ext.DaoImplVWS  
2 metier.MetierImpl  
3
```

The bottom screenshot shows the 'Run' console for 'App2'. It displays the command executed and the output:

```
Run: "C:\Program Files\Java\jdk1.8.0_301\bin\java.exe" ...  
Version Web Services :  
40.0  
Process finished with exit code 0
```

Nous venons ainsi de faire l'injection des dépendances par instanciation statique puis par instanciation dynamique.